TEST PROCEDURES
for
HIGH LEVEL ARCHITECTURE
INTERFACE SPECIFICATION

14 May 1997
VERSION 1.1

Prepared by:

Margaret L. Loper, Margaret M. Horst
Georgia Tech Research Institute
Georgia Institute of Technology
margaret.loper@gtri.gatech.edu, margaret.horst@gtri.gatech.edu

TABLE OF CONTENTS

1. OVERVIEW

This document contains test procedures for the High Level Architecture Interface Specification, v1.1. The remainder of this section discusses terminology used within this document, organization of the procedures, and how procedures are defined.

1.1 DEFINITIONS

The following definitions apply through this document:

Action Services HLA Interface Services used by the RTI and federate to exchange information in order to accomplish an action.

Black Box Testing A testing process in which the internal design of the Federate Under Test is not visible from the outside.

Conformance Testing The process of verifying that a federate performs in accordance with the interface specification. This is not the same as compliance testing.

Control Services HLA Interface services used by the RTI to initialize federates.

Federate Under Test (FUT) The federate containing the Implementation Under Test.

Federation Execution Data (FED) Information derived from the FOM (class, attribute, parameter names, etc.) and used by the RTI at run time. Each federation execution needs one.

Federation Object Model (FOM) An identification of the essential classes of objects, object attributes, and object interactions that are supported by an HLA federation.

Initialization Services HLA Interface services used by federates to initialize the RTI.

Management Services HLA Interface Services used to control the operation of the federation or to request information about the state of a federation.

RTI Initialization Data (RID) RTI vendor specific information needed to run an RTI. A RID is probably supplied when an RTI is initialized.

Simulation Object Model (SOM) A specification of the intrinsic capabilities that an individual simulation offers to federations.

State Transition Diagrams (STD) A method for specifying the finite state machine of each interface service. The STD is described in terms of state variables.

State Variables A set of variables that describe the state of the federate (or RTI) prior to or as a result of invoking an interface service.

1.2 ORGANIZATION OF TEST PROCEDURES

The services contained in the Interface Specification are at an individual-level, each individual service is described in terms of its inputs, outputs and exceptions. There is only partial guidance for how the individual services are used together or the order in which they should be invoked. The test procedures provide more guidance by creating an execution framework (or recommended order) in which the interface services should be used. This is only guidance; there is no requirement to conform to this framework.

Test procedures are based on both individual-level and functional-level services. For the purpose of these procedures, functional-level services are collections of individual-level services used for a specific function. Individual- and functional-level services are organized into four categories (Initialization, Control, Action, and Management), which form the execution framework, as shown in Figure 1. Each category is described below.

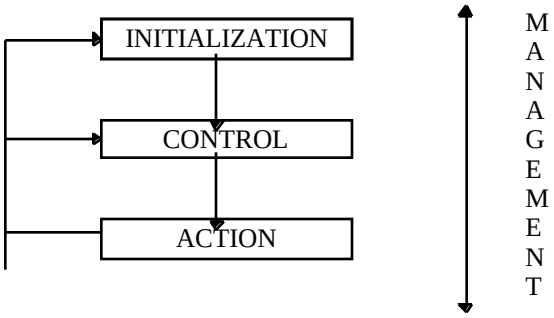

Figure 1: Execution Framework

Initialization Services are used by federates to set state in the RTI. These services typically follow a posting paradigm, in that the federate tells the RTI what it is capable or what it requires for execution. Initialization services are at the individual-level and are shown in Table 1.

Create/Destroy Federation
Join/Resign Federation
Subscribe (Object Class Attribute, Interaction Class)
Publish (Object Class, Interaction Class)
Object Instantiation (Request Id, Register Object)
Request Federation Time
Set Look Ahead
Create Update/Subscription Region, Associate Update Region

Table 1: Initialization Services

Control Services are used by the RTI to initialize federates. These services also follow a posting paradigm in that the RTI tells the federate what publications and interactions it is allowed to update during the execution. Control services are also at an individual-level and are contained in Table 2.

Control Updates
Control Interactions
Discover Object
Change Thresholds

Table 2: Control Services

Action Services are used by the RTI and federate to exchange information in order to accomplish an action. These services follow a request/response paradigm in that the federate requests some action to be completed and the RTI responds with confirmation or notification that the action has taken place. Action services are at the functional-level and are contained in Table 3.

Update/Reflect Attribute
Send/Receive Interaction
Request/Provide Attribute

Modify Region
Change Attribute Transportation Type, Order Type
Change Interaction Transportation Type, Order Type
Delete/Remove Object
Attribute Ownership Acquisition
Attribute Ownership Divestiture
Query Attribute Ownership, Inform Attribute Ownership
Is Attribute Owned by Federate
Time Advance, Next Event

Table 3: Action Services

Management Services follow a request/response paradigm and are used to control the operation of the federation or to request information about the state of a federation. Management services can be invoked at any time after Initialization; all management services have a pre-condition that a Federation Execution exists and that the Federate is a member of the Execution. Management services are both individual- and functional-level services and are shown in Table 4.

Pause/Resume Federation Execution
Save/Restore Federation Execution
Request Federation Time, Federate Time
Request LBTS, Minimum Next Event Time
Request Lookahead
Flush Queue Request

Table 4: Management Services

1.3 State Transition Diagrams and State Variables

The test procedures contained in this document are based on state transition diagrams (STDs), which is a common method for specifying communication protocols. For each procedure, the individual- or functional-level services are defined in terms of a set of states, the inputs and their effect on the states, and the corresponding outputs. Thus, the state diagrams model the fact that a given input will cause the protocol entity (i.e., the part of the federate that implements the service) to go from one state to another, and possibly also cause a particular output to occur. The state diagrams are present purely for descriptive purposes: they serve as a model of reality.

The procedures in this document include tests for initiating and responding to the interface services. Since it is hard to decouple the federates actions from those of the RTI (in terms of the procedures), the STDs reflect the behavior of both.

For all federate initiated services (except Create Federation Execution, Destroy Federation Execution and Join Federation Execution), there is an implied supplied parameter which is a federation execution. For all RTI initiated services, there is an implied supplied parameter which is a federate. The manner in which these parameters are actually provided to the services is RTI implementation dependent, and therefore not shown in the STDs.

In defining the STDs, a set of variables that describe the state of the federate (and RTI) prior to or as a result of invoking an interface service were developed. These variables are called State Variables. The state variables are listed within each STD and are based on the HLA Interface Specification, not the Interface Definition Language (IDL) Application Programmer Interface (API). The State Variables defined for testing are listed in Table 5.

At this time, there is no standard way to access the state variables for the purpose of testing the Interface Specification. The state variables are presented purely for descriptive purposes.

Federation (designator, FED, exist/not exist, members)
Federate (designator, connection parameters, state, routing space designator)
Pause(pausing/running, label, time)
Save(saving/restoring, label, time, paused, achieved)
Query(request, question, who, result)
Object (class designator, id, exist/not exist, attribute designator, attribute value, ownership, removal reason, subscription reason designator, update region designator, extents)
Interaction (class designator, parameters, id)
Time (federation, federate, logical, tag, lookahead, transport data ordering, designator, #events, offset, scalar, Minimum Next Event Time, LBTS)
ID (requested, number returned, federate limit, federation limit)
Capability (publish, include/exclude flag, subscribe, flag, predicates)
Divest (time, reason, status)
Assumption (time, reason, priority)
Acquisition (time, reason, release time, release reason)

Table 5: State Variables

There are various entities (classes, attributes, parameters, regions, federates, etc.) referenced in this document which can have the following different “views”:

- name - human readable
- handle - computer manipulable

The parameters to the services described in this document will use different views of the entities depending on a particular RTI implementation. For clarity, this document refers only to a generic view, known as a “designator,” when referring to these entities.

2. FEDERATE TESTING

This section describes the following issues surrounding federate testing: test set-up, test methods, and performing the test.

2.1 TEST SET-UP

To conduct a test for the interface specification, two federates are required. The first federate is the Federate Under Test (FUT). The FUT is the federate which contains the implementation being tested. The second federate is the "Tester." The term "Tester" is used generically to refer to the person or tool conducting the test. Since there are no test tools currently available for interface testing, the Test Federate supplied with the RTI is considered the "Tester." The FUT and Test Federate are connected via the RTI, as shown in Figure 2.

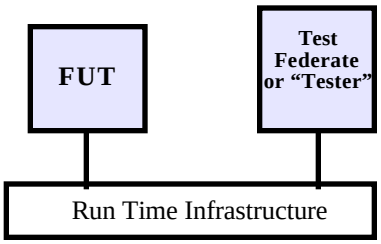

Figure 2: Test Configuration

For Interface testing a federate relies on information in the SOM. This phase of testing does not rely on a federation. However, in order to use the configuration described in Figure 2 and because all interface services (except Create Federation) have a prerequisite that a Federation exists, both the FUT and Test Federate must join a federation. Therefore, for the purpose of federate interface testing the following assumptions are made: the FOM is the same as the SOM and the FED is generated from the SOM.

2.2 TEST METHODS

2.2.1 Black Box Testing

The test set-up described in Section 2.1 is called black box testing. This means that the "Tester" cannot access the internal workings of the implementation of the FUT. Therefore, all the "Tester" can do is invoke services and see how the FUT reacts. This type of testing evaluates the order (and behavior) of services, it does not examine syntax and semantics. Currently, correct syntax is implied from successful interoperation with the RTI. Correct semantics is determined by invoking services according to the execution framework until such time a problem occurs. If a problem is found, the human "Tester" must consider which services are prerequisites for the service that failed and determine what is the likely problem. The black box testing approach described above is a generic process that can be used throughout the test process.

2.2.2 White Box Testing

Another test method that can be used is white box testing. In this approach, the "Tester" has access to internal data from the FUT or can examine the FUT's code during testing. Since tools do not currently exist for this method of testing, a test process cannot be described. At some future date, the state variables described in section 1.3 may be used for observing the internal state

of a FUT. At that time a test process describing their use by the "Tester" will be included in this section.

2.3 PERFORMING THE TEST

The first step in performing interface testing is to determine the services to be tested. This is accomplished by documenting a FUT's capabilities in a federate Conformance Statement (CS). The CS stands as a record of which services have been tested for each FUT. The CS is included in Appendix A

The next step of conducting a test is to determine the data to be used for testing. A Test Sequence is constructed based on the Scenario Data submitted by the federation (if available). If the federate developer chooses not to submit Scenario Data, a Test Sequence is arbitrarily created based on the SOM and CS. The federate developer will review the Test Sequence generated by the Certification Agent, will download and install the RTI to use for testing at the developer's site, and will submit test environment data to the Certification Agent. The IF test is executed by the federate developer and the Certification Agent.

The IF Test has two parts: the Nominal test, which ensures that the FUT can invoke and respond to all services for which it is capable, per its CS; and the Representative Som (RepSOM) test, which ensures that the FUT is capable of invoking and responding to services using the range of data contained in its SOM. The Certification Agent will log service data from the test, analyze the data, generate results, and return a Certification Summary Report (CSR) to the federate developer. The CSR is the official record of HLA compliance for the specific version of the federate code tested.

3. ASSUMPTIONS

The following assumptions apply through this document:

1. Federate test procedures presume that the RTI used during testing conforms to the HLA interface specification. Conformance testing of the RTI is outside the scope of this document.
2. Conformance testing excludes any assessment of performance, robustness, or reliability. Only correctness of the implementation is under test, not its speed of operation nor efficiency of its operation, nor the optimization of its code, nor other running capabilities.
3. This documents does not contain procedures for stress testing, invalid or adverse data, or out of bound conditions. These tests are important for conformance testing and should be accomplished operationally.
4. The exception behavior specified in the STD is one example of how a federate can react. There is no requirement to conform to this.
5. Test procedures defined in this document are based solely on the Interface Specification; test procedures are not based on the IDL API.

4. INITIALIZATION AND CONTROL SERVICES

Initialization services are used by federates to set state in the RTI and control (update, interaction) services are used by the RTI to control publication in the federate. These services are shown together in the test procedures since the RTI's control is a response to what the federate initializes.

4.1 CREATE FEDERATION EXECUTION

The Create Federation Execution function is used to build a new federation execution. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		No federation exists	None
Operating State		Wait for services to be issued	None
Event	Federation Manager issues Create Federation Execution	Check Registrations for valid Federation(name, FED); on error go to operating state	Submit Federation(name, FED) to the RTI
Post Conditions		A federation exists with the given name that can be joined by federates	

4.1.1 Traceability

Section 2.1 Create Federation Execution

4.1.2 Initiating Create Federation Execution

The FUT shall invoke the Create Federation Execution service with the supplied parameters:

Federation Execution Name: Test
FED: Selected by FUT

4.2 DESTROY FEDERATION EXECUTION

The Destroy Federation Execution function is used to remove the named federation execution from the set of supported federation executions. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Federation exists	Resigned from Federation Execution
Operating State		Wait for services to be issued	None
Event	Federation Manager issues Destroy Federation Execution	Check Registrations for Federation(members, exist); on error go to operating state	Submit Federation(name) to the RTI
Post Conditions		Register Federation(not exist)	

4.2.1 Traceability

Section 2.2 Destroy Federation Execution

4.2.2 Initiating Destroy Federation Execution

The FUT shall invoke the Destroy Federation Execution service with the supplied parameters:

Federation Execution Name: Test

4.3 JOIN FEDERATION EXECUTION

The Join Federation Execution function is used to affiliate the federate with the federation execution and declare important properties about the federate. The STD is below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows members of federation	None
Operating State		Knows services issued	None
Event	Federate issues Join Federation Execution	Check Registrations for valid Federate(designator) and Federation(name, members); on error go to operating state	Submit Federate(designator) Federation(name), and optional connection parameters to the RTI
Post Conditions		Federate is member of named Federation Execution	Federate is member of named Federation Execution

4.3.1 Traceability

Section 2.3 Join Federation Execution

4.3.2 Initiating Join Federation Execution

The FUT shall invoke the Join Federation Execution service with the supplied parameters:

Federate Designator: supplied by FUT
Federation Execution Name: Test
Connection Parameters: tbd/optional

The RTI shall return the parameters:

Federate Designator	*selected by RTI*

4.4 RESIGN FEDERATION EXECUTION

The Resign Federation Execution function is used to cease the participation of the federate in the named federation execution. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership	Member of Federation
Operating State		Knows services issued	Knows current state
Event	Federate issues Resign Federation Execution	Check Registrations for valid Federate(designator) and Federation(member); perform optional directive if necessary; on error go to operating state	Submit optional directive t the RTI. If the optional directive is not supplied, the federate should not own any attributes.
Post Conditions		There are no attributes in the federation execution owned by the specified federate.	Federate not a member of Federation Execution

4.4.1 Traceability

Section 2.4 Resign Federation Execution

4.4.2 Initiating Resign Federation Execution

The FUT shall invoke the Resign Federation service.

4.4.3 Initiating Resign Federation Execution (Optional Parameters)

The FUT shall invoke the Resign Federation service with the supplied parameters:

Optional directive:	(1) release all attribute ownership, (2) delete all objects for which the federate holds delete privilege, (3) perform action (2) and then action (1), or (4) perform no actions

4.5 SUBSCRIBE OBJECT CLASS ATTRIBUTE

The Subscribe Object Class Attribute function is used to control which classes of objects the federate will discover and which attribute value the federate will receive from that class. This function is to be called iteratively by the FUT to subscribe to multiple attributes of an object class. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Knows current state
Event	Federate issues Subscribe Object Class Attribute	Check Registrations for valid Federate(designator) and Federation(member); check FED for valid Object(class, attribute designator), Capability(include/unsubscribe); verify object is associated with region; on error go to operating state	Submit Object(class designator), Capability(include/exclude flag, region) and set of attribute designators (subscribe only) to RTI
Post Conditions		The RTI has been informed of th federate's desired subscription or unsubscription.	

4.5.1 Traceability

Section 3.3 Subscribe Object Class Attribute

4.5.2 Initiating Subscribe Object Class Attribute

The FUT shall invoke the Subscribe Object Class Attribute service with the supplied parameters:

Object Class Designator: selected from FED
Include/Exclude Option: selected by FUT
if Include,
Attribute Designator: selected from FED

4.5.3 Initiating Subscribe Object Class Attribute (Optional Parameters)

The FUT shall invoke the Subscribe Object Class Attibute service with the supplied parameters:

Object Class Designator: selected from FED
Include/Exclude Option: selected by FUT
if Include,
Attribute Designators: selected from FED
Region: optional

4.6 Subscribe Interaction Class

The Subscribe Interaction Class function is used to specify the classes of interactions which should or should not be sent to the federate. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Knows current state
Event	Federate issues Subscribe Interaction Class	Check Registrations for valid Federate(designator) and Federation(member); check FED for valid Interaction(class designator); on error go to operating state	Submit Interaction(class designator), Capability(sub, include/exclude flag) to the RTI with optional subscriptic region designator
Post Conditions		Register Capability(subscribe, include/exclude flag) for Interaction(class)	

4.6.1 Traceability

Section 3.4 Subscribe Interaction Class

4.6.2 Initiating Subscribe Interaction Class

The FUT shall invoke the Subscribe Interaction class service with the supplied parameters:

Interaction Class Designator:	selected from FED
Include/Exclude Flag:	selected by FUT (include, exclude)

4.6.3 Initiating Subscribe Interaction Class (Optional Parameters)

The FUT shall invoke the Subscribe Interaction class service with the supplied parameters:

Interaction Class Designator:	selected from FED
Include/Exclude Flag:	selected by FUT (include,exclude)
Region:	optional

4.7 PUBLISH OBJECT CLASS

The Publish Object Class function is used to indicate which classes of objects the federate is capable of providing to the federation and specifies the attributes to be supplied for those object classes. The Control Updates function is used to tell the federate that the specified attributes for the specified class or object are or are not required somewhere in the federation. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Knows current state
Event 1	Federate issues Publish Object Class	Check Registrations for valid Federate(designator) and Federation(member); check FED for valid Object(class designator, attribute designators); on error go to operating state	Submit Object(class designator, attribute designators), Capability(publish, include/exclude flag) to RTI
Post Conditions		Register Capability(publish, include/exclude flag) for Object(attribute designators)	
Event 2	RTI issues Control Updates	Submit Object(class designator, attribute designators) and Capability(update/no update) to the federate	Check Registrations for valid published Object(attribute designators); on error go to operating state
Post Conditions			Register Capability(update/no update) for Object(attribute designators)

4.7.1 Traceability

Section 3.1 Publish Object Class
Section 3.5 Control Updates

4.7.2 Initiating Publish Object Class

The FUT shall invoke the Publish Object Class service with the supplied parameters:

Object Class Designator: selected from FED
Attribute Designators: selected from FED
Include/Exclude Flag: selected by FUT (include,exclude)

The RTI shall invoke the Control Updates service to the FUT with the supplied parameters:

Object Class Designator: supplied by RTI
Attribute Designators: supplied by RTI
Update Flag: selected by RTI (update, no update)

4.8 PUBLISH INTERACTION CLASS

The Publish Interaction Class function is used to specify which classes of interactions the federate will be publishing to the federation. The Control Interactions function is used to tell the federate that the specified class of interactions is or is not required somewhere in the federation. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Knows current state
Event 1	Federate issues Publish Interaction Class	Check Registrations for valid Federate(designator) and Federation(member); check FED for valid Interaction(class designator); on error go to operating state	Submit Interaction(class designator), Capability(publish, include/exclude flag) to the RTI
Post Conditions		Register Capability(publish, include/exclude flag) for Interaction(class designator)	

4.8.1 Traceability

Section 3.2 Publish Interaction Class
Section 3.6 Control Interactions

4.8.2 Initiating Publish Interaction Class

The FUT shall invoke the Publish Interaction Class service with the supplied parameters:

Interaction Class Designator: selected from FED
Include/Exclude Flag: selected by FUT (include, exclude)

The RTI shall invoke the Control Interactions service to the FUT with the supplied parameters:

Interaction Class Designator: supplied by FUT
Publish Flag: selected by RTI (publish, not publish)

4.9 REQUEST ID

The Request Id function is used to request federation execution-unique object ID numbers. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Knows current state
Event	Federate issues ID Request	Check Registrations for valid Federate(designator), Federation(member), and number of IDs that are not reused or reserved; on error go to operating state	Submit ID(requested) to the RTI
Post Conditions		Return ID(numbers) to the federate	

4.9.1 Traceability

Section 4.1 Request Id

4.9.2 Initiating Request Id

The FUT shall invoke the Request Id service with the supplied parameters:

Number of Ids:	selected by FUT

The RTI shall return the parameters:

Set of Ids:	*selected by RTI*

4.10 REGISTER OBJECT

The Register Object function is used to link an object Id with an instance of an object class. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Publishing object class
Event	Federate issues Instantiate Object	Check FED for valid Object(clas designator); check Registrations for valid object(id), Capability (publish, include/exclude flag); on error go to operating state.	Submit Object(class designator, id), to the RTI
Post Conditions		Register Object(id, ownership)	Register Object(id, ownership)

4.10.1 Traceability

Section 4.2 Register Object

4.10.2 Initiating Register Object

The FUT shall invoke the Register Object service with the supplied parameters:

Object Class Designator:	selected from RTI
Object ID:	previously returned from RTI

4.11 SET LOOKAHEAD

The Set Look Ahead function is used to set the desired value of the federate's lookahead. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	
Event	Federate issues Set Lookahead	Check Registrations for Federate(designator), Federation(member), and for valid Time(lookahead); on error go to operating state	Submit Time(lookahead) to RTI
Post Conditions		Register Time(lookahead) for Federate(designator)	

4.11.1 Traceability

Section 6.5 Set Lookahead

4.11.2 Initiating Set Lookahead

The FUT shall invoke the Set Look Ahead service with the supplied parameters:

Lookahead: selected by FUT

4.12 CREATE UPDATE REGION

The Create Update Region function is used to create a subset of the specified routing space. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	
Event	Federate issues Create Update Region	Check Registrations for valid Federate(designator), Federation(member), check FED for valid Federate(Routing Space Designator, Extents); on error go to operating state.	Submit Federate(Routing Space Designator, Extents) to the RTI
Post Conditions		Return Update Region Handle	Register Federate(Routing Space Designator, Extents, Update Region Designator)

4.12.1 Traceability

Section 7.1 Create Update Region

4.12.2 Initiating Create Update Region

The FUT shall invoke the Create Update Region service with the supplied parameters:

Routing Space Designator Selected from FED

Extents Selected by FUT

The RTI shall return the parameter:

Update Region Designator: selected by RTI

4.13 CREATE SUBSCRIPTION REGION

The Create Subscription Region function is used to create a subset of the specified routing space. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	
Event	Federate issues Create Update Region	Check Registrations for valid Federate(designator), Federation(member), check FED for valid Federate(Routing Space Designator, Extents); on error go to operating state.	Submit Federate(Routing Space Designator, Extents) to the RTI
Post Conditions		Return Subscription Region Designator	Register Federate(Routing Space Designator, Extents, Subscription Region Designator)

4.13.1 Traceability

Section 7.2 Create Subscription Region

4.13.2 Initiating Create Subscription Region

The FUT shall invoke the Create Subscription Region service with the supplied parameters:

Routing Space Designator Selected from FED
Extents Selected by FUT

The RTI shall return the parameter:

Subscription Region Designator: selected by RTI

4.14 ASSOCIATE UPDATE REGION

The Associate Update Region service associates an update region with attributes of a specific object or an interaction class. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Object(Id, Attributes) is registered, and being published, and update region exists.
Event	Federate issues Associate Update Region	Check FED for valid Object(class); check Registrations for valid Federate(designator), Federation(member), object(id, update region designator), Capability (publish, include/exclude flag); on error go to operating state.	Submit Object(update region designator, association instruction, Id, Attributes) to the RTI
Post Conditions		Register Object(id, update region)	Register Object(id, ownership)

4.14.1 Traceability

Section 7.3 Associate Update Region

4.14.2 Initiating Associate Update Region for Attributes of an Object

The FUT shall invoke the Associate Update Region service with the supplied parameters:

Update Region Designator	returned from RTI in 4.11
Object ID:	returned from RTI in 4.8
Attributes:	selected by FUT
Association Instruction	(form, break)

4.14.3 Initiating Associate Update Region for Interaction Classes

The FUT shall invoke the Associate Update Region service with the supplied parameters:

Update Region Designator	returned from RTI in 4.11
Interaction Class:	selected by FUT
Association Instruction	(form, break)

4.15 DELETE REGION

The Delete Region function is used delete the specified update or subscription region. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Update/subscription region exists.
Event	Federate issues Delete Region	Check Registrations for valid Federate(designator), Federation(member), Object(Update/Subscription Designator); on error go to operating state.	Submit Object(Update/Subscription Designator) to the RTI
Post Conditions		Delete Object(Update/Subscription Region)	

4.15.1 Traceability

Section 7.6 Delete Region

4.15.2 Initiating Delete Region

The FUT shall invoke the Delete Region service with the supplied parameters:

Update/Subscription Region Designator returned from RTI in 4.11

5. ACTION AND CONTROL SERVICES

Action services are used by the federate and RTI to exchange information in order to accomplish an action for the federate. The control service in this section is used by the RTI to inform the federate of an object which requires reflection.

5.1 DISCOVER OBJECT

The Discover Object function is used to inform the federate that the RTI has discovered an object in the federation when the following occur: 1) An attribute update is received from another federate, 2)The federate has neither registered nor discovered the object, 3)
The update satisfies the federate's description, and 4) The update is in the associated region (if regions are being used.) This service is issued by the RTI as a response to the Update Attribute Value services. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STA
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation	Member of Federation
Operating State		Knows publication and subscription capabilities of federates; waiting for services to be issued	Publishing object class	Subscribed to object class
Event 1	Federate A issues Update Attribute Values	Check FED for valid (attribute designator, attribute value); check Registrations for valid Object(id, ownership); check for valid Time(federation,tag) on error go to operating state	Submit Object(id, attribute designator, attribute value), Time(federation, tag) to the RTI	
Post Conditions		Return Time(designator) to federate; Search Registrations for valid federate(designator), federation(member), Object(id, attribute designator, attribute value) with Capability(subscribe, include/exclude flag)		
Event 2	RTI issues Discover Object to Federate B	Submit Object(class, id), Time(federation,designator, tag) to the federate		Check Registrations for valid subscribed Object(id); check for valid Time(federation, tag); on error go to operating state
Post Conditions			Register reflected Object(id)	

5.1.1 Traceability

Section 4.3 Update Attribute Values
Section 4.4 Discover Object

5.1.2 Responding to Discover Object

The Test Federate shall invoke the Update Attribute Values service with the supplied parameters:

Object Id:	returned from RTI in 4.8
Attribute Designator:	selected from FED
Attribute Value:	selected by Test Federate
Federation Time:	selected by Test Federate
User Supplied Tag:	selected by Test Federate

The RTI shall return the parameter:

Event Retraction Designator:	*selected by RTI*

The RTI shall invoke the Discover Object service to the FUT with the supplied parameters:

Object ID:	*supplied by Test Federate*
Object Class Designator:	*selected by RTI*
Federation Time:	*supplied by Test Federate*
User Supplied Tag:	*selected by Test Federate*
Retraction Designator:	*selected by RTI*

5.2 UPDATE/REFLECT ATTRIBUTES

The Update/Reflect Attributes functions are used to provide the current attribute values to the federation. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation	Member of Federation
Operating State		Wait for services to be issued	Publishes object attributes	Subscribes to object attributes
Event 1	Federate A issues Update Attribute Values	Check FED for valid (attribute designator, attribute value);, check Registrations for valid Federate(designator), Federation(member), Object(id, ownership); check for valid Time(federate, tag) on error go to operating state	Submit Object(id, attribute designator, attribute value), Time(federate, tag) to the RTI	
Post Conditions		Return Time(designator) to federate; Search Registrations for valid federate(designator), federation(member), Object(id, attribute designator, attribute value) with Capability(subscribe, include/exclude flag)		
Event 2	RTI issues Reflect Attribute Values to Federate B	Submit Object(id, attribute designator, attribute value) and Time(federation, designator) to federates registered for Capability(sub, inc/exc flag)		Check Registrations for valid Object(id); check FED for valid Object(attribute designator, attribute value); check for valid Time(federation, timestamp); on error go to operating state
Post Conditions		The new attribute values have been supplied to the federate		

5.2.1 Traceability

Section 4.3 Update Attribute Values
Section 4.5 Reflect Attribute Values

5.2.2 Initiating Update Attribute Values

The FUT shall invoke the Update Attribute Values service with the supplied parameters:

Object Id: returned from RTI in 4.8

Attribute Designators, Values: selected by FUT
Federate Time: selected by FUT
User Supplied Tag: selected by FUT

The RTI shall return the parameter:

Event Retraction Designator: selected by RTI

The RTI shall invoke the Reflect Attribute Values service to the Test Federate with the supplied parameters:

Object Id: selected by FUT
Attribute Designators, Values: selected by FUT
Federation Time: selected by FUT
User Supplied Tag: selected by Test Federate
Retraction Designator: selected by RTI

5.2.3 Responding to Update Attribute Values

The Test Federate shall invoke the Update Attribute Value service with the supplied parameters:

Object Id: returned from RTI in 4.8
Attribute Designators, Values: selected by Test Federate
Federate Time: selected by Test Federate
User Supplied Tag: selected by Test Federate

The RTI shall return the parameter:

Event Retraction Designator: selected by RTI

The RTI shall invoke the Reflect Attribute Value service to the FUT with the supplied parameters:

Object Id: supplied by Test Federate
Attribute Designators, Values: supplied by Test Federate
Federation Time: supplied by Test Federate
User Supplied Tag: selected by Test Federate
Retraction Designator: selected by RTI

5.3 SEND/RECEIVE INTERACTION

The Send/Receive Interaction function is used to inform the federation of an action taken by one object towards another object. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STA
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation and publishing interaction class	
Operating State		Knows publication and subscription capabilities of federates; waiting for services to be issued.	Publishing Interaction Class	Subscribing to Interaction Class
Event 1	Federate A issues Send Interaction	Check FED for valid Interaction(class designator, parameters); check Registrations for federate(designator), federation(member); check for valid Time(federation); on error go to operating state	Submit Interaction(class designator, parameters), and Time(federation,tag) to RTI	
Post Conditions		RTI returns Time(designator) to federate		
Event 2	RTI issues Receive Interaction to Federate B	Submit Interaction(class, parameters), and Time(federation,tag) to the federate		Check FED for valid Interaction(class, parameters); check Registrations for valid subscription Interaction(class); check for valid Time(federation); on error go to operating state
Post Conditions				

5.3.1 Traceability

Section 4.6 Send Interaction
Section 4.7 Receive Interaction

5.3.2 Initiating Send Interaction

The FUT shall invoke the Send Interaction service with the supplied parameters:

Interaction Class Designator:	selected from FED
Parameter Designators, Values:	selected from FED
Federation Time:	selected by FUT
User Supplied Tag:	selected by FUT

The RTI shall return the parameters:

Retraction Designator: *selected by RTI*

The RTI shall invoke the Receive Interaction service to the Test Federate with the parameters:

Interaction Class Designator: *supplied by FUT*
Interaction Parameter Designators, Values: *supplied by FUT*
Federation Time: *supplied by FUT*
User Supplied Tag: *selected by FUT*
Retraction Designator: *selected by RTI*

5.3.3 Responding to Send Interaction

The Test Federate shall invoke the Send Interaction service with the supplied parameters:

Interaction Class Designator: selected from FED
Parameter Designators, Values: selected from FED
Federate Time: selected by Test Federate
User Supplied Tag: selected by FUT

The RTI shall return the parameters:

Retraction Designator: *selected by RTI*

The RTI shall invoke the Receive Interaction service to the FUT with the parameters:

Interaction Class Designator: *supplied by Test Federate*
Interaction Parameter Designators, Values: *supplied by Test Federate*
Federation Time: *supplied by Test Federate*
User Supplied Tag: *selected by Test Federate*
Retraction Designator: *selected by RTI*

5.4 REQUEST/PROVIDE ATTRIBUTE

The Request/Provide Attribute function is used to request the current attribute values for the specified object or object class from the federation member responsible for publishing this information. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation	Member of Federation
Operating State		Wait for services to be issued	Subscribing to object attributes	Publishing object attribu
Event 1	Federate A issues Request Attribute Value Update	federate(designator, federation(member), valid Object(id); check FED for valid Object(attribute designator); on err go to operating state	Submit Object(id, attribute designator) to the RTI	
Post Conditions		Check Registrations for Object(id) vs Federate(id) to locate federate publishing attributes		
Event 2	RTI issues Provide Attribute Value Update to Federate B	Submit Object(id, attribute designator) to the federate		Check FED for valid Object(attribute designator); check Registrations for valid Object(id); on error go to operating state
Post Conditions				
Event 3	Federate B issues Update Attribute Values	Check FED for valid (attribute designator, attribute value); check Registrations for federate(designator), federation(member) and valid Object(id, ownership); check for valid Time(federation); on error go to operating state		Submit Object(id, attribut designator, attribute valu and Time(federation, tag) to the RTI
Post Conditions		Return Time(designator) to federate B; Search Registions for Object(id, attribute designator, attribute value with Capability(subscribe, include/exclude flag)		
Event 4	RTI issues Reflect Attribute Values	Submit Object(id, attribute designator, attribute value) and Time(federation, tag) to federates Registered for Capability(subscrib include/exclude flag)	Check Registrations for valid Object(id); check FE for valid Object(attribute designator, attribute value check for valid Time(federation, tag); on error go to operating state	
Post Conditions				

5.4.1 Traceability

Section 4.14 Request Attribute Value Update
Section 4.15 Provide Attribute Value Update

Section 4.3 Update Attribute Values
Section 4.5 Reflect Attribute Values

5.4.2 Initiating Request Attribute Value

The FUT shall invoke the Request Attribute Value Update service with the supplied parameters:

Object Id or Object Class:	selected by FUT
Attribute Designators:	selected by FUT

The RTI shall invoke the Provide Attribute Value Update service to the Test Federate with the parameters:

Object Id:	*supplied by FUT or RTI*
Attribute Designators:	*supplied by FUT*

The Test Federate shall invoke the Update Attribute Values service with the supplied parameters:

Object Id:	selected by FUT or RTI
Attribute Designators:	selected by FUT
Attribute Value:	selected by Test Federate
Federate Time:	selected by Test Federate
User Supplied Tag:	selected by Test Federate

The RTI shall return the parameter:

Event Retraction Designator:	*selected by RTI*

The RTI shall invoke the Reflect Attribute Values service to the FUT with the supplied parameters:

Object Id:	*selected by FUT or RTI*
Attribute Designators, Values	*selected by Test Federate*
Federation Time:	*selected by Test Federate*
User Supplied Tag:	*selected by Test Federate*
Retraction Designator:	*selected by RTI*

5.4.3 Responding to Request Attribute Value

The Test Federate shall invoke the Request Attribute Value Update service with the supplied parameters:

Object Id or Object Class:	selected by Test Federate
Attribute Designators:	selected by Test Federate

The RTI shall invoke the Provide Attribute Value Update service to the FUT with the parameters:

Object Id:	*selected by Test Federate*
Attribute Designators:	*selected by Test Federate*

The FUT shall invoke the Update Attribute Values service with the supplied parameters:

Object Id:	selected by Test Federate or RTI
Attribute Designators, Values:	selected by FUT

Federation Time: selected by FUT
User Supplied Tag: selected by FUT

The RTI shall return the parameter:

Event Retraction Designator: selected by RTI

The RTI shall invoke the Reflect Attribute Value service to the Test Federate with the supplied parameters:

Object Id: selected by Test Federate or RTI
Attribute Designators, Values: selected by FUT
Federation Time: supplied by FUT
User Supplied Tag: selected by FUT
Retraction Designator: selected by RTI

5.5 CHANGE ATTRIBUTE TRANSPORTATION TYPE

The Change Attribute Transportation Type service is used to change the transportation type for the specified attributes on the specified object.

	SERVICE	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	Owns attribute(s)
Event 1	Federate issues Change Attribute Transportation Type	Check for Federate(designator), Federate(member); check Registrations for ownership Object(Id, Attribute Set), check for valid Time(transport service); on error go to operatin state	Submit Object(Id, Attribute Set), Time(transportation service) to RTI
Post Conditions		Register Time(transportation service)	Register Time(transportation service)

5.5.1 Traceability

Section 4.10 Change Attribute Transportation Type

5.5.2 Initiating Change Attribute Transportation Type

The FUT shall invoke the Change Attribute Transportation Type service with the supplied parameters:

Object Id: selected from FED
Attribute Designators: set of attributes of object
Transportation Type: tbd

5.6 CHANGE ATTRIBUTE ORDER TYPE

The Change Attribute Order Type service is used to change the data ordering type for the specified attributes on the specified object.

	SERVICE	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	Owns attribute(s)
Event 1	Federate issues Change Attribute Order Type	Check for Federate(designator), Federate(member); check Registrations for ownership Object(Id,Attribute Set), Time(Data Ordering Type); on error go to operating state	Submit Object(Id, Attribute Set) Time(Data Ordering Type) to RTI,
Post Conditions		Register Time(Data Ordering Type)	Register Time(Data Ordering Type)

5.6.1 Traceability

Section 4.11 Change Attribute Order Type

5.6.2 Initiating Change Attribute Order Type

The FUT shall invoke the Change Attribute Order Type service with the supplied parameters:

Object Id:	selected from FED
Attribute Designators:	set of attributes of object
Data Ordering Type:	tbd

5.7 CHANGE THRESHOLDS

The Change Thresholds service provides the federate with the new values of the thresholds for each of the dimensions of a routing space. The new threshold data should be used to decide when to use the Modify Region service. The Modify Region service is used to change the bounds of the update or subscription to reflect the specified set of extents. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Wait for services to be issued	Object(Id, Attributes) is registered, and being published, and update region exists.
Event	Federate issues Associate Update Region	Check FED for valid Object(class); check Registrations for valid Federate(designator), Federation(member), object(id, update region designator), Capability (publish, include/exclude flag); on error go to operating state.	Submit Object(update region designator, association instruction, Id, Attributes) to the RTI
Post Conditions		Register Object(id, update region)	Register Object(id, ownership)

5.7.1 Traceability

Section 7.4 Change Thresholds
Section 7.5 Modify Region

5.7.2 Initiating Change Thresholds

The RTI shall invoke the Change Thresholds service with the supplied parameters:

Update/Subscription Region Designator:	*selected by RTI*
Thresholds:	*selected by RTI*

The FUT shall invoke the Modify Region service with the supplied parameters:

Update/Subscription Region Designator:	returned from RTI in 4.11
Extents:	selected by FUT

5.8 CHANGE INTERACTION TRANSPORTATION TYPE

The Change Interaction Transportation Type service is used to change the transportation type for the specified interaction class.

	SERVICE	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	Publishing interaction
Event	Federation Manager issues Change Federate Time	Check for Federate(designator), Federate(member); check Registrations for publishing of Interaction(class), Time(Transportation Type); on error go to operating state	Submit Interaction(class) Time(Transportation Type) to RTI.
Post Conditions		Register Time(Transportation Type) for Federate(name)	Register Time(Transportation Type)

5.8.1 Traceability

Section 4.12 Change Interaction Transportation Type

5.8.2 Initiating Change Interaction Transportation Type

The FUT shall invoke the Change Interaction Transportation Type service with the supplied parameters:

Interaction Class:	selected from FED
Transportation Type:	tbd

5.9 CHANGE INTERACTION ORDER TYPE

The Change Interaction Order Type service is used to change the data ordering type for the specified Interaction class.

	SERVICE	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	Publishing interaction
Event	Federation Manager issues Change Federate Time	Check for Federate(designator), Federate(member); check Registrations for publishing of Interaction(class designator), Time(Data Ordering Type); on error go to operating state	Submit Interaction(class designator), Time(Data Ordering Type) to RTI.
Post Conditions		Register Interaction(Data Ordering Type) for Federate(designator);	Register Interaction(Data Ordering Type)

5.9.1 Traceability

Section 4.13 Change Interaction Order Type

5.9.2 Initiating Change Interaction Order Type

The FUT shall invoke the Change Interaction Order Type service with the supplied parameters:

Interaction Class:	selected from FED
Data Ordering Type:	tbd

5.10 DELETE/REMOVE OBJECT

The Delete/Remove Object function is used to inform the federation that an object is to be removed from the federation. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STA
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation	Member of Federation
Operating State		Knows publication and subscription capabilities of federates; waiting for services to be issued	Owns the object ID attribute	Reflects the object ID attribute
Event 1	Federate A issues Delete Object	Check Registrations for vali Federate(designator), Federation(member), Object(id, ownership); check for valid Time(federation, tag); on error go to operatin state	Submit Object(id) and Time(federation, tag) to the federate	
Post Conditions		Return Time(designator); Register Object(not exist)	Register Capability(publish, include/exclude flag) for Object(id)	
Event 2	RTI issues Remove Object to Federate B	Submit Object(id,removalreason) and Time(federation, tag) to federates subscribed to Object(id)		Check Registrations for valid Object(id); check for valid Time(federation); on error go to operating stat
Post Conditions				Register Object(not exist)

5.10.1 Traceability

Section 4.8 Delete Object
Section 4.9 Remove Object

5.10.2 Initiating Delete Object

The FUT shall invoke the Delete Object service with the supplied parameters:

Object Id:	selected by FUT
Federation Time:	selected by FUT
User Supplied Tag:	selected by FUT

The RTI shall return the following parameter:

Retraction Designator:	*selected by RTI*

The RTI shall invoke the Remove Object service to the Test Federate with the supplied parameters:

Object Id:	*supplied by FUT*
Object Removal Reason:	*(deleted, out-of-scope)*
Federation Time:	*supplied by FUT*
User Supplied Tag:	*selected by FUT*

5.10.3 Initiating Delete Object (Optional Parameters)

The FUT shall invoke the Delete Object service with the supplied parameters:

Object Id:	selected by FUT
Federation Time:	selected by FUT
User Supplied Tag:	selected by FUT

The RTI shall return the following parameter:

Retraction Designator:	*selected by RTI*

The RTI shall invoke the Remove Object service to the Test Federate with the supplied parameters:

Object Id:	*supplied by FUT*
Object Removal Reason:	*(deleted, out-of-scope)*
Federation Time:	*supplied by FUT*
User Supplied Tag:	*selected by FUT*
Retraction Designator:	*supplied by FUT/optional*

5.10.4 Responding to Delete Object

The Test Federate shall invoke the Delete Object service with the supplied parameters:

Object Id:	selected by Test Federate
Federation Time:	selected by Test Federate
User Supplied Tag:	selected by Test Federate

The RTI shall return the following parameter:

Retraction Designator:	*selected by RTI*

The RTI shall invoke the Remove Object service to the FUT with the supplied parameters:

Object Id:	*supplied by Test Federate*
Object Removal Reason:	*(deleted, out-of-scope)*
Federation Time:	*supplied by Test Federate*

User Supplied Tag: *selected by Test Federate*

5.10.5 Responding to Delete Object (Optional Parameters)

The Test Federate shall invoke the Delete Object service with the supplied parameters:

Object Id:	selected by Test Federate
Federation Time:	selected by Test Federate
User Supplied Tag:	selected by Test Federate

The RTI shall return the following parameter:

Retraction Designator:	*selected by RTI*

The RTI shall invoke the Remove Object service to the FUT with the supplied parameters:

Object Id:	*supplied by Test Federate*
Object Removal Reason:	*(deleted, out-of-scope)*
Federation Time:	*supplied by Test Federate*
User Supplied Tag:	*selected by Test Federate*
Retraction Designator:	*supplied by Test Federate/optional*

5.11 Attribute Ownership Acquisition

The Attribute Ownership Acquisition function is used to request privilege to own the specified attributes of the specified objects. The RTI will request that the federate owning the attributes release ownership, and then notify the initiating federate of the result. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE A STATE	FEDERATE B STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation	Member of Federation
Operating State		Waiting for services to be issued	Invoked Publish Object Class or Publish Object Attributes; Owns specified object attributes	Invoked Publish Object Class or Publish Object Attributes
Event 1	Federate B issues Request Attribute Ownership Acquisitio	Check Registrations for valid Object(id, attribute designators); check for valid Federation(member) Federate(designator) for Capability(publish); che val Acquisition(time) >= Time(federation); check ? for valid time; on error go to operating state		Submit Object(id, attribute designators), Time(tag) to RTI
Post Conditions		Search Registrations for Federate(designator) of object(id, ownership, attribute designators)		
Event 2	RTI issues Request Attribute Ownership Release to Federate A	Submit Object(id, attribut designators) and Time(tag to federate	Check Registrations for valid Object(id, attribute designators); on error go t operating state	
Post Conditions		Register Object (id, attribute designators, ownership) for Federate(null)	Return Object(id, attribute designators); Register release of Object (id, attribute designator, ownership)	
Event 3	RTI issues Attribute Ownership Acquisition Notification to Federa B	Submit Object(id, attribut designators) to federate		Check Registrations for valid Object(id, attribute designators); on error go t operating state
Post Conditions		Register Object (id, ownership, attribute designators) for		Register Object (id, ownership, attribute designators)

5.11.1 Traceability

Section 5.5 Request Attribute Ownership Acquisition
Section 5.6 Request Attribute Ownership Release
Section 5.4 Attribute Ownership Acquisition Notification

5.11.2 Initiating Attribute Ownership Acquisition

The FUT shall invoke the Request Attribute Ownership Acquisition service with the supplied parameters:

Object Id:	selected by FUT
Attribute Designators:	selected by FUT
User Supplied Tag:	selected by FUT

The RTI shall invoke the Request Attribute Ownership Release to Test Federate with the parameters:

Object Id:	*selected by FUT*
Attribute Designators	*selected by FUT*
User Supplied Tag:	*selected by FUT*

The Test Federate shall return the parameters:

Attribute Designators:	selected by Test Federate

The RTI shall invoke the Attribute Ownership Acquisition Notification to the FUT with parameters:

Object Id:	*selected by FUT*
Attribute Designators:	*returned from Test Federate*

5.11.3 Responding to Attribute Ownership Acquisition

The Test Federate shall invoke the Request Attribute Ownership Acquisition service with the supplied parameters:

Object Id:	selected by Test Federate
Attribute Designators:	selected by Test Federate
User Supplied Tag:	selected by Test Federate

The RTI shall invoke the Request Attribute Ownership Release to the FUT with the parameters:

Object Id:	*selected by Test Federate*
Attribute Designators:	*selected by Test Federate*
User Supplied Tag:	*selected by Test Federate*

The FUT shall return the parameters:

Attribute Designators:	selected by FUT

The RTI shall invoke the Attribute Ownership Acquisition Notification to the Test Federate with parameters:

Object Id:	*selected by Test Federate*
Attribute Designators:	*returned from FUT*

5.12 ATTRIBUTE OWNERSHIP DIVESTITURE

The Attribute Ownership Divestiture function is used to tell the RTI that the federate no longer wants to own the specified attributes of the specified objects. The RTI will look for a new owner for the attributes and notify the initiating and responding federates of the result. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation	Member of Federation
Operating State		Wait for services to be issued	Owns specified attributes or object IDs	Invoked Publish Object Class or Publish Object Attributes
Event 1	Federate A issues Request Attribute Ownership Divestiture	Check Registrations for valid Federate(designator), Federation(member), Object(id), and Object (id, ownership); check FED for valid Object(attribute designators); on error go to operating state	Submit Object(id, attribute designators, divestiture condition), Time(tag), option Federate(designators) to RTI	
Post Conditions		If Federate(designator) not specified, check Registrations for federates with Capability(publish)		
Event 2	RTI issues Request Attribute Ownershi Assumption to Federate B	Submit Object(id, attribute designators)		Check Registrations for valid Object(id, attribute designators); on error go to operating state
Post Conditions				Return Object(attribute designators)
Event 3	RTI issues Attribute Ownership Acquisition Notification to Federate B	Submit Object(id, attribute designators) to federate		Check Registrations for valid Object(id, attribute designators); on error go to operating state
Post Conditions				Register Object (id, ownershi attribute designators)
Event 4	RTI issues Attribute Ownership Divestiture Notification to Federate A	Submit Object(ids, attribut designators) to federate	Check Registrations for valid Object(id, attribute designators, ownership); on error go to operating state.	
Post Conditions		Register Object (id, ownership, attribute designators) with Federate(id)	Register release of Object (id ownership, attribute designators)	

5.12.1 Traceability

Section 5.1 Request Attribute Ownership Divestiture
Section 5.2 Request Attribute Ownership Assumption
Section 5.4 Attribute Ownership Acquisition Notification
Section 5.3 Attribute Ownership Divestiture Notification

5.12.2 Initiating Attribute Ownership Divestiture

The FUT shall invoke the Request Attribute Ownership Divestiture service with the supplied parameters:

Object Id:	selected by FUT
Attribute Designators:	selected by FUT
Ownership Divestiture Condition:	negotiated
User Supplied Tag:	selected by FUT

The RTI shall invoke the Request Attribute Ownership Assumption to the Test Federate service with the supplied parameters:

Object Id:	*supplied by FUT*
Attribute Designators:	*supplied by FUT*
User Supplied Tag:	*supplied by FUT*

The Test Federate shall return the parameters:

Attribute Designators:	selected from those supplied by RTI

The RTI shall invoke the Attribute Ownership Acquisition Notification to Test Federate with the supplied parameters:

Object Id:	*supplied by Test Federate*
Attribute Designators:	*supplied by Test Federate*

The RTI shall invoke the Attribute Ownership Divestiture Notification to the FUT with the supplied parameters:

Object Ids:	*supplied by Test Federate*
Attribute Designators:	*supplied by Test Federate*

5.12.3 Initiating Attribute Ownership Divestiture (Optional Parameters)

The FUT shall invoke the Request Attribute Ownership Divestiture service with the supplied parameters:

Object Id:	selected by FUT
Attribute Designators:	selected by FUT
Ownership Divestiture Condition:	unconditional
User Supplied Tag:	selected by FUT
Federate Designator:	Test Federate

The RTI shall invoke the Request Attribute Ownership Assumption service to the Test Federate with the supplied parameters:

Object Id: *supplied by FUT*
Attribute Designators: *supplied by FUT*
User Supplied Tag: *supplied by FUT*

The Test Federate shall return the parameters:

Attribute Designators: selected from those supplied by RTI

The RTI shall invoke the Attribute Ownership Acquisition Notification to the Test Federate with the supplied parameters:

Object Id: *supplied by Test Federate*
Attribute Designators: *supplied by Test Federate*

The RTI shall invoke the Attribute Ownership Divestiture Notification service to the FUT with the supplied parameters:

Object Ids: *supplied by Test Federate*
Attribute Designators: *supplied by Test Federate*

5.12.4 Responding to Attribute Ownership Divestiture

The Test Federate shall invoke the Request Attribute Ownership Divestiture service with the supplied parameters:

Object Id: selected by Test Federate
Attribute Designators: selected by Test Federate
Ownership Divestiture Condition: negotiated
User Supplied Tag: selected by Test Federate

The RTI shall invoke the Request Attribute Ownership Assumption service to the FUT with the supplied parameters:

Object Id: *supplied by Test Federate*
Attribute Designators *supplied by Test Federate*
User Supplied Tag: *supplied by Test Federate*

The FUT shall return the parameters:

Attribute Designator: selected from those supplied by RTI

The RTI shall invoke the Attribute Ownership Acquisition Notification to the FUT with the supplied parameters:

Object Id: *supplied by FUT*
Attribute Designators: *supplied by FUT*

The RTI shall invoke the Attribute Ownership Divestiture Notification to Test Federate service with the supplied parameters:

Object Ids: *supplied by FUT*
Attribute Designator: *supplied by FUT*

5.12.5 Responding to Attribute Ownership Divestiture (Optional Parameters)

The Test Federate shall invoke the Request Attribute Ownership Divestiture service with the supplied parameters:

Object Id:	selected by Test Federate
Attribute Designators:	selected by Test Federate
Ownership Divestiture Condition:	unconditional
User Supplied Tag:	selected by FUT
Federate Designator:	FUT

The RTI shall invoke the Request Attribute Ownership Assumption service to the FUT with the supplied parameters:

Object Id:	*supplied by Test Federate*
Attribute Designators:	*supplied by Test Federate*
User Supplied Tag:	*supplied by Test Federate*

The FUT shall return the parameters:

Attribute Designator:	selected from those supplied by RTI

The RTI shall invoke the Attribute Ownership Acquisition Notification to the FUT with the supplied parameters:

Object Id:	*supplied by FUT*
Attribute Designators:	*supplied by FUT*

The RTI shall invoke the Attribute Ownership Divestiture Notification service to Test Federate with the supplied parameters:

Object Ids:	*supplied by FUT*
Attribute Designators:	*supplied by FUT*

5.13 Query Attribute Ownership

The Query Attribute Ownership function is used to determine if the specified attributes of the specified object Ids are owned and if so by which federate. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	
Event 1	Federate issues Query Attribute Ownership	Check Registrations for valid Object(id, attribute designator); on error go to operating state	Submit Object(id, attribute designator) to RTI
Event 2	RTI issues Inform Attribute Ownership	Return Federate(designator) and Object(id, attribute) or none	
Post Conditions			The federate has been informed of the attribute ownership

5.13.1 Traceability

Section 5.7 Query Attribute Ownership
Section 5.8 Inform Attribute Ownership

5.13.2 Initiating Query Attribute Ownership

The FUT (as FM) shall invoke the Query Attribute Ownership service with the supplied parameters:

Object Id:	supplied by FUT
Attribute Designator:	selected from FED

The RTI shall invoke the Inform Attribute Ownership service with the parameters:

Object ID:	*supplies by FUT*
Attribute Designator:	*selected from FED*
Federate Designator:	*designator or "none"*

5.14 Is Attribute Owned by Federate

The Is Attribute Owned by Federate function is used to determine if the specified attributes of the specified object Ids are owned by the invoking federate. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	
Event	Federate issues Query Attribute Ownership	Check Registrations for valid federate(designator), federation(member), Object(id, attribute designator); on error go to operating state	Submit Object(id, attribute designator) to RTI
Post Conditions		Return Federate(designator) or none	

5.14.1 Traceability

Section 5.9 Is Attribute Owned by Federate

5.14.2 Initiating Is Attribute Owned by Federate

The FUT shall invoke the Is Attribute Owned by Federate service with the supplied parameters:

Object Id:	supplied by FUT
Attribute Designator:	selected from FED

The RTI shall return the parameters:

Attribute ownership designator:	*Boolean*

5.15 TIME ADVANCE FUNCTION

The Time Advance function requests and advance of the federate's logical time. Invocation of this service implies that the following messages are eligible for delivery to the federate: (1) all incoming receive ordered messages, and (2) all messages using other ordering services with timestamp less than or equal to t. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	Federate is internally consistent and ready to mov to specified logical time; Subscribed to interaction class and/or Attribute Value
Event 1	Federate issues Time Advance Request	Check Registrations for Federate(designator), Federation(member), check for valid Time(logical); on error go to operating state	Submit Time(logical) to RT
Post Conditions		If events eligible go to E2 & deliver message to federate; if no eligible messages, go to E3	
Event 2	RTI issues Receive Interaction (or Reflect Attribute)	Submit Interaction(class designator, parameter designators, values), and Time(federation, tag, handle) to the federate	Check FED for valid Interaction(class, designators, parameters); check Registrations for subscribed Interactions; on error go to operating state
Post Conditions		For TSO: Time(logical) = Time(federation); if non-TSO: Time(logical) = Time(logical)	
Event 3	RTI issues Time Advance Grant	Submit Time(federation) to federate	Check Registrations for valid Time(logical); on error go to operating state
Post Conditions		Register Time(logical) for Federate(designator)	

5.15.1 Traceability

Section 6.7 Time Advance Request
Section 6.10 Time Advance Grant

5.15.2 Initiating Time Advance

The FUT shall invoke the Time Advance Request service with the supplied parameters:

Logical Time: supplied by FUT

The RTI shall invoke the Receive Interaction service to the FUT with the parameters:

Interaction Class Designator: *supplied by Test Federate*
Interaction Parameter Designators, Values: *supplied by Test Federate*
Federation Time: *supplied by Test Federate*
User Supplied Tag: *supplied by FUT*
Retraction Designator: *supplied by RTI*

The RTI shall invoke the Time Advance Grant service with the supplied parameters:

Federation Time: *supplied by RTI*

5.15.3 Responding to Time Advance

No separate test required.

5.16 NEXT EVENT

The Next Event function is used to request the next time stamp ordered (TSO) message from the RTI, provided that message has a time stamp no greater than the logical time specified in the request. The invocation of this service implies that the following messages are eligible for delivery to the federate: (1) all receive ordered messages, (2) the smallest time stamped TSO message that will ever be delivered in the future with time stamp less than or equal to the specified time, and all other TSO messages containing this same time stamp value, and (3) all messages using other ordering services with time stamp less than or equal to the specified time. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	Federate is internally consistent and ready to mov to specified logical time; Subscribed to interaction class and/or Attribute Value
Event 1	Federate issues Next Event Request	Check for valid Time(logical) on error go to operating state	Submit Time(logical, #events) to RTI
Post Conditions		If events eligible go to E2 & deliver message to federate; if no eligible messages, go to E	
Event 2	RTI issues Receive Interaction (or Reflect Attribute)	Submit Interaction(class, parameter designators, values) and Time(federation, tag, handle) to the federate	Check FED for valid Interaction(class designators, parameters); check Registrations for subscribed Interactions; on error go to operating state
Post Conditions		For TSO: Time(logical) = Time(federation); if non-TSO: Time(logical) = Time(logical)	
Event 3	RTI issues Time Advance Grant	Submit Time(federation) to federate	Check Registrations for valid Time(logical); on erro go to operating state
Post Conditions		Register Time(logical) for Federate(designator)	

5.16.1 Traceability

Section 6.8 Next Event Request
Section 6.10 Time Advance Grant

5.16.2 Initiating Next Event

The FUT shall invoke the Next Event Request service with the supplied parameters:

Federate Time: selected by FUT

The RTI shall invoke the Receive Interaction service to the FUT with the parameters:

Interaction Class Designator: *supplied by Test Federate*
Interaction Parameter Designators, Values: *supplied by Test Federate*
Federation Time: *supplied by Test Federate*
User Supplied Tag: *supplied by FUT*
Retraction Designator: *supplied by RTI*

The RTI shall invoke the Time Advance Grant service with the supplied parameters:

Federation Time: *supplied by RTI*

5.16.3 Responding to Next Event

No separate test required.

5.17 RETRACT

The Retract function is used to retract a previously scheduled event. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	Federate has previously issued Update Attribute Values and Send Interaction service calls and obtained retraction designators
Event 1	Federate issues Retract	Check Registrations for valid federate(designator), federation(member), and Time(designators); on error go to operating state	Submit Time(handles) to RTI
Post Conditions		RTI checks buffers for retracted event; if buffered, retract and go to operating state, else go to E2	
Event 2	RTI issues Reflect Retraction	Submit Time(designator) to federates	Check Registrations for valid Time(designators)
Post Conditions			Cancel effects of Interaction(designator)

5.17.1 Traceability

Section 4.16 Retract
Section 4.17 Reflect Retraction
Section 4.3 Update Attribute Values
Section 4.6 Send Interaction

5.17.2 Initiating Retract

The FUT shall invoke the Retract service with the supplied parameters:

Retraction Designator:	supplied by RTI in 4.3 (Update Attribute Values) or 4.6 (Send Interaction)

The RTI shall invoke the Reflect Retraction service to the Test Federate with the supplied parameters:

Retraction Designator:	*supplied by FUT*

5.17.3 Responding to Retract

The Test Federate shall invoke the Retract service with the supplied parameters:

Retraction Designator:	supplied by RTI in4.3 (Update Attribute Values) or 4.6 (Send Interaction)

The RTI shall invoke the Reflect Retraction service to the FUT with the supplied parameters:

Retraction Designator:	*supplied by Test Federate*

6. MANAGEMENT SERVICES

Management services are used to control the operation of the federation or to request information about the state of a federation.

6.1 PAUSE FEDERATION EXECUTION

The Pause Federation Execution function is used to stop the advance of the federation. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STAT
Pre-Conditions		Knows federation membership	Member of Federation	Member of Federation
Operating State		Knows services issued	Federation not paused	Federation not paused
Event 1	Federation Manager issues Request Pause	Check Registrations for Federate(designator), Federation(member), Pause(pausing); on error go to operating state	Submit Pause(label) to RTI	
Post Conditions		Register Pause(label); previous Pause invalidated by new registration	Federate is running	
Event 2	RTI issues Initiate Pause to Federate B	Submit Pause(label) to federate		Check Registrations for Pause(pausing); on error go operating state
Post Conditions				Start Pause(pausing); previous Pause invalidated by new registration
Event 3	Federate B issues Pause Achieved	Register Pause(label);on error go to operating state;		Submit Pause(label) to RTI
Post Conditions				Register Pause(paused)

6.1.1 Traceability

Section 2.5 Request Pause
Section 2.6 Initiate Pause
Section 2.7 Pause Achieved

6.1.2 Initiating Pause Federation

The FUT (as FM) shall invoke the Request Pause service with the supplied parameters:

Pause Label: Interface

The RTI shall invoke the Initiate Pause service to Test Federate with the supplied parameters:

Pause Label: *Interface*

The Test Federate shall invoke the Pause Achieved service with the supplied parameters:

Pause Label: Interface

6.1.3 Responding to Pause Federation

The Test Federate (as FM) shall invoke the Request Pause service with the supplied parameters:

Pause Label: Interface

The RTI shall invoke the Initiate Pause service to the FUT with the supplied parameters:

Pause Label: *Interface*

The FUT shall invoke the Pause Achieved service with the supplied parameters:

Pause Label: Interface

6.2 Resume Federation Execution

The Resume Federation Execution function is used to resume advance of the federation. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STATE
Pre-Conditions		Knows federation membership	Member of federation	Member of federation
Operating State		Knows services issued	Federate is Paused	Federate is Paused
Event 1	Federation Manager issues Request Resume	Check Registrations for Federate(designator), Federation(member), Pause(pausing); on error go to operating state		
Post Conditions				
Event 2	RTI issues Schedule Resume	Submit to the federate		Check Registrations for Pause(pausing); on error g to operating state
Post Conditions				Start Restore
Event 3	Federate issues Resume Achieved	on error go to operating state		Submit to the RTI
Post Conditions				Register Pause(running)

6.2.1 Traceability

Section 2.8 Request Resume
Section 2.9 Initiate Resume
Section 2.10 Resume Achieved

6.2.2 Initiating Resume Federation

The FUT (as FM) shall invoke the Request Resume service.

The RTI shall invoke the Initiate Resume service to Test Federate.

The Test Federate shall invoke the Resume Achieved.

6.2.3 Responding to Resume Federation

The Test Federate (as FM) shall invoke the Request Resume service.

The RTI shall invoke the Initiate Resume service to FUT.

The FUT shall invoke the Resume Achieved service.

6.3 SAVE FEDERATION EXECUTION

The Save Federation Execution function is used to save the state of the federation. A rolling save takes place as the federation continues to advance. A paused save takes place after the federation has been paused. The STD is shown below.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STA
Pre-Conditions		Knows federation membership	Member of federation	Member of federation
Operating State		Knows services issued	Knows current state	Knows current state
Event	Federation Manager issues Request Federate Save	Check Registrations for Federate(designator) and Federation(member); on error go to operating state	Submit Save(label) to the RTI	
Post Conditions		Register Save(label); cancels previously scheduled saves;		
Event 2	RTI issues Initiate Federate Save to Federate B	Submit Save(label) to federate		on error go to operating state
Post Conditions				
Event 3	Federate B issues Federate Save Begun	Check Registrations for Save(saving); on error g to operating state		Submit to RTI
Post Conditions		Have all federates issued Save Begun? If no, go t E2.		Register Save(saving)
Event 4	Federate B issues Federate Save Achieved	on error go to operating state		Submit Save(achieved) to RTI
Post Conditions				

6.3.1 Traceability

Section 2.11 Request Federation Save
Section 2.12 Initiate Federate Save
Section 2.13 Federate Save Begun
Section 2.14 Federate Save Achieved

6.3.2 Initiating Save Federation

The FUT (as FM) shall invoke the Request Federation Save service with the supplied parameters:

Save Label:	Interface

The RTI shall invoke the Initiate Federate Save service to Test Federate with the supplied parameters:

Save Label:	*Interface*

The Test Federate shall invoke the Federate Save Begun service.

The Test Federate shall invoke the Federate Save Achieved service with the supplied parameters:

Save Success Indicator:	supplied by Test Federate

6.3.3 Initiating Save Federation (Optional Parameter)

The FUT (as FM) shall invoke the Request Federation Save service with the supplied parameters:

Save Label:	Interface
Save Time:	selected by FUT/optional

The RTI shall invoke the Initiate Federate Save service to Test Federate with the supplied parameters:

Save Label:	*Interface*
Save Time:	*selected by FUT/optional*

The Test Federate shall invoke the Federate Save Begun service with the supplied parameters:

Save Time:	selected by FUT/optional

The Test Federate shall invoke the Federate Save Achieved service with the supplied parameters:

Save Success Indicator:	supplied by Test Federate

6.3.4 Responding to Save Federation

The Test Federate (as FM) shall invoke the Request Federation Save service with the supplied parameters:

Save Label:	Interface

The RTI shall invoke the Initiate Federate Save service to FUT with the supplied parameters:

Save Label:	*Interface*

The FUT shall invoke the Federate Save Begun service.

The FUT shall invoke the Federate Save Achieved service with the supplied parameters:

Save Success Indicator:	supplied by FUT

6.3.5 Responding to Save Federation (Optional Parameter)

The Test Federate (as FM) shall invoke the Request Federation Save service with the supplied parameters:

Save Label:	Interface
Save Time:	selected by Test Federate/optional

The RTI shall invoke the Initiate Federate Save service to FUT with the supplied parameters:

Save Label:	*Interface*
Save Time:	*selected by Test Federate/optional*

The FUT shall invoke the Federate Save Begun service with the supplied parameters:

Save Time:	selected by Test Federate/optional

The FUT shall invoke the Federate Save Achieved service with the supplied parameters:

Save Success Indicator:	supplied by FUT

6.4 RESTORE FEDERATION EXECUTION

The Restore Federation Execution function is used to restore the federation to a previously saved state.

	SERVICE	RTI STATE	FEDERATE A STATE	FEDERATE B STATE
Pre-Conditions		Knows federation membership	Member of federation	Member of federation
Operating State		Knows services issued	Knows current state	Knows current state
Event 1	Federation Manager issues Request Restore	Check Registrations for Federate(designator), Federation(member); check for valid Save(label); on error go to operating state	Submit Save(label) to RTI	
Post Conditions				
Event 2	RTI issues Initiate Restore to Federate B	Submit Save(label) to federates		Check Registrations for valid Save(label); on error go to operating state
Post Conditions				Register Save(restoring)
Event 3	Federate B issues Restore Achieved	Check Registrations for valid Save(restoring, label); on error go to operating state		Submit Save(Achieved, label) to RTI
Post Conditions				Federate is in state identical to that when Save was issued

6.4.1 Traceability

Section 2.15 Request Restore
Section 2.16 Initiate Restore
Section 2.17 Restore Achieved

6.4.2 Initiating Restore Federation

The FUT (as FM) shall invoke the Request Restore service with the supplied parameters:

Save Label: Interface

The RTI shall invoke the Initiate Restore service to the Test Federate with the supplied parameters:

Save Label: *Interface*

The Test Federate shall invoke the Restore Achieved service with the supplied parameters:

Save Label: Interface

6.4.3 Responding to Restore Federation

The Test Federate (as FM) shall invoke the Request Restore service with the supplied parameters:

Save Label: Interface

The RTI shall invoke the Initiate Restore service to the FUT with the supplied parameters:

Save Label: Interface

The FUT shall invoke the Restore Achieved service with the supplied parameters:

Save Label: Interface

6.5 REQUEST FEDERATION TIME

The Request Federation Time function is used to request the current estimate of the federation time. Federation time is the minimum of the Lower Bound Time Stamp (LBTS) and the current value of federate's Logical Time (LT). The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Estimate of Federation time is available; waiting for services to be issued	
Event	Federate issues Request Federation Time	Check that Time(federation) is available; on error go to operating state	
Post Conditions		Return Time(federation)	

6.5.1 Traceability

Section 6.1 Request Federation Time

6.5.2 Initiating Request Federation Time

The FUT shall invoke the Request Federation Time service.

The RTI shall return the parameters:

Federate Time: (current minimum of LBTS and LT)

6.6 REQUEST LBTS

The LBTS function is used to request the current value of the Lower Bound Time Stamp (LBTS). The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Time(LBTS) is available; waiting for services to be issued	
Event	Federate issues Request LBTS	Check that Time(LBTS) is available); on error go to operating state	
Post Conditions		Return Time(LBTS)	

6.6.1 Traceability
Section 6.2 Request LBTS

6.6.2 Initiating Request LBTS
The FUT shall invoke the Request LBTS service.

The RTI shall return the parameters:

Federate Time: *(current value of LBTS)*

6.7 REQUEST FEDERATE TIME
The Request Time function is used to request the current value of the federate's Logical Time. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Logical time has been set; waiting for services to be issued	
Event	Federate issues Request Federate Time	Check that Time(federate) has been set; on error go to operating state	
Post Conditions		Return Time(federate)	

6.7.1 Traceability
Section 6.3 Request Federate Time

6.7.2 Initiating Request Federate Time
The FUT shall invoke the Request Time service.

The RTI shall return the parameters:

Federate Time: *(current value of LT)*

6.8 REQUEST MINIMUM NEXT EVENT TIME

The Request Minimum Next Event Time service is used to request the minimum of LBTS and the time stamp of the next Time Stamp Ordered (TSO) message that is held by the RTI for delivery to the requesting federate.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Time(Minimum Next Event Time) is available; waiting for services to be issued	
Event	Federate issues Request Federate Time	Check that Time(Minimum Next Event Time) is available; on error go to operating state	
Post Conditions		Return Time(Minimum Next Event Time)	

6.8.1 Traceability

Section 6.4 Request Minimum Next Event Time

6.8.2 Initiating Request Minimum Next Event Time

The FUT shall invoke the Request Minimum Next Event Time service.

The RTI shall return the parameters:

Federate Time: *(current minimum of LBTS and the head of the TSO queue)*

6.9 REQUEST LOOKAHEAD

The Request Lookahead function is used to request the current value of lookahead for the federate. The current value of the lookahead may differ temporarily from the desired lookahead given in the Set Lookahead service if the value of the lookahead has been reduced. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Federate lookahead has been set; waiting for services to be issued	
Event	Federate issues Request Lookahead	Check that Time(lookahead) has been set; on error go to operating state	
Post Conditions		Return Time(lookahead)	

6.9.1 Traceability

Section 6.6 Request Lookahead

6.9.2 Initiating Request Lookahead

The FUT shall invoke the Request Lookahead service.

The RTI shall return the parameters:

Lookahead:	*federate's current value of lookahead*

6.10 Flush Queue Request

The Flush Queue Request function releases all messages stored in the RTI's internal queues and delivers them to the federate. The STD is shown below.

	SERVICE ISSUED	RTI STATE	FEDERATE STATE
Pre-Conditions		Knows federation membership and RTI initialization content	Member of Federation
Operating State		Waiting for services to be issued	
Event	Federate issues Flush Queue Request	Check that Time(federation) is valid; on error go to operating state	Submit Time(federation) to RTI
Post Conditions		Check Registrations for pending messages for federate(designator); on error go to operating state	
Event 2	RTI issues Receive Interaction to Federate	Submit Interaction(class, parameters), and Time(federation,tag) to the federate	Check FED for valid Interaction(class, parameters); check Registrations for valid subscription Interaction(class); check for valid Time(federation); on error go to operating state
Post Conditions			
Event 3	RTI issues Time Advance Grant	Submit Time(federation) to federate	Check Registrations for valid Time(federation); on error go to operating state
Post Conditions		Register Time(logical) for Federate(designator)	

6.10.1 Traceability

Section 6.9 Flush Queue Request
Section 4.7 Receive Interaction
Section 6.10 Time Advance Grant

6.10.2 Initiating Flush Queue Request

The FUT shall invoke the Flush Queue Request service with the supplied parameters:

Federation Time: selected by FUT

The RTI shall invoke the Receive Interaction service to the FUT with the parameters:

Interaction Class Designator: supplied by FUT
Interaction Parameter Designators, Values: supplied by FUT
Federation Time: supplied by FUT
User Supplied Tag selected by FUT
Event Retraction Designator: selected by RTI

The RTI shall invoke the Time Advance Grant service with the supplied parameters:

Federation Time: supplied by RTI

7. APPENDIX A: FEDERATE CONFORMANCE STATEMENT

FEDERATE CONFORMANCE STATEMENT

INTERFACE SPECIFICATION V1.1

31 March 1997

SERVICE GROUP	SERVICE	IF Ref	OMT Ref	Check List	M/O	Supported ?
Create/Destroy	Create Federation Execution	2.1	None	Item 6	M	Yes
	Destroy Federation Execution	2.2	None	Item 6	M	Yes
Join/Resign	Join Federation Execution	2.3	None	Item 6	M	Yes
	Resign Federation Execution	2.4	None	Item 6	M	Yes
Pause/Resume	Request Pause	2.5	None	Item 6	O	Yes No
	Initiate Pause†	2.6	None	Item 6	O	Yes No
	Paused Achieved	2.7	None	Item 6	O	Yes No
	Request Resume	2.8	None	Item 6	O	Yes No
	Initiate Resume†	2.9	None	Item 6	O	Yes No
	Resume Achieved	2.10	None	Item 6	O	Yes No
Save/Restore	Request Federation Save	2.11	None	Item 6	O	Yes No
	Initiate Federate Save†	2.12	None	Item 6	O	Yes No
	Federation Save Begun	2.13	None	Item 6	O	Yes No
	Federation Save Achieved	2.14	None	Item 6	O	Yes No
	Request Restore	2.15	None	Item 6	O	Yes No
	Initiate Restore†	2.16	None	Item 6	O	Yes No
	Restore Achieved	2.17	None	Item 6	O	Yes No
Publication and Subscription	Publish Object Class	3.1	3.1, 3.2, 3.3	Item 2	O	Yes No
	Subscribe Object Class Attributes	3.3	3.1, 3.3	Item 2	O	Yes No
	Publish Interaction	3.2	3.2	Item 2	O	Yes No
	Subscribe Interaction	3.4	3.2	Item 2	O	Yes No
Flow Control	Control Updates†	3.5	None	Item 2	O	Yes No
	Control Interactions†	3.6	None	Item 2	O	Yes No
Object Representation	Request ID	4.1	None	Item 6	O	Yes No
	Register Object	4.2	3.1	Item 6	O	Yes No
	Update Attribute Values	4.3	3.2, 3.3	Item 2&4	O	Yes No

	Discover Object†	4.4	3.1	Item 6	O	Yes No
	Reflect Attribute Values†	4.5	3.3	Item 2&4	O	Yes No
	Delete Object	4.8	None	Item 6	O	Yes No
	Remove Object†	4.9	None	Item 6	O	Yes No
Object Interactions	Send Interaction	4.6	3.2	Item 2	O	Yes No
	Receive Interaction†	4.7	3.2	Item 2	O	Yes No
Attribute Transport and Ordering	Change Attribute Transportation Type	4.10	None	Item 6	O	Yes No
	Change Attribute Order Type	4.11	None	Item 6	O	Yes No
Interaction Transport and Ordering	Change Interaction Transportation Type	4.12	None	Item 6	O	Yes No
	Change Interaction Order Type	4.13	None	Item 6	O	Yes No

Request Attribute Values	Request Attribute Value Update	4.14	None	Item 2	O	Yes	No
	Provide Attribute Value Update†	4.15	None	Item 2	O	Yes	No
Event Retraction	Retract	4.16	None	Item 2	O	Yes	No
	Reflect Retract†	4.17	None	Item 2	O	Yes	No
Ownership Divestiture	Request Attribute Ownership Divestiture	5.1	3.3	Item 3	O	Yes	No
	Request Attribute Ownership Assumption†	5.2	3.3	Item 3	O	Yes	No
	Attribute Ownership Divestiture Notification†	5.3	3.3	Item 3	O	Yes	No
	Attribute Ownership Acquisition Notification†	5.4	3.3	Item 3	O	Yes	No
Ownership Acquisition	Request Attribute Ownership Acquisition	5.5	3.3	Item 3	O	Yes	No
	Request Attribute Ownership Release†	5.6	3.3	Item 3	O	Yes	No
	Attribute Ownership Acquisition Notification†	5.4	3.3	Item 3	O	Yes	No
Ownership Query	Query Attribute Ownership	5.7	None	Item 3	O	Yes	No
	Inform Attribute Ownership†	5.8	None	Item 3	O	Yes	No
	Is Attribute Owned by Federate	5.9	None	Item 3	O	Yes	No
Request Time	Request Federation Time	6.1	None	Item 5	O	Yes	No
	Request LBTS	6.2	None	Item 5	O	Yes	No
	Request Federate Time	6.3	None	Item 5	O	Yes	No
	Request Min Next Event	6.4	None	Item 5	O	Yes	No

	Time						
Lookahead	Set Lookahead	6.5	None	Item 5	O	Yes	No
	Request Lookahead	6.6	None	Item 5	O	Yes	No
Advance Request	Time Advance Request	6.7	None	Item 5	O	Yes	No
	Time Advance Grant†	6.10	None	Item 5	O	Yes	No
Next Event Request	Next Event Request	6.8	None	Item 5	O	Yes	No
	Time Advance Grant†	6.10	None	Item 5	O	Yes	No
Flush Queue Request	Flush Queue Request	6.9	None	Item 5	O	Yes	No
	Time Advance Grant†	6.10	None	Item 5	O	Yes	No

REFERENCES

F. Halsall, Data Communications, Computer Networks and Open Systems, Third Edition, Addison-Wesley Publishing Company, 1992.

B. Zeigler, Theory of Modelling and Simulation, Robert E. Krieger Publishing Company, 1984.

K. Knightson, OSI Protocol Conformance Testing: IS 9646 Explained, McGraw-Hill, Inc., 1993.